Fast Numerical Methods for Bernoulli Free Boundary Problems

Christopher M. Kuster, Pierre A. Gremaud, Rachid Touzani

FAST NUMERICAL METHODS FOR BERNOULLI FREE BOUNDARY PROBLEMS

CHRISTOPHER M. KUSTER*, PIERRE A. GREMAUD†, AND RACHID TOUZANI‡

Abstract. The numerical solution of the free boundary Bernoulli problem is addressed. An iterative method based on a level-set formulation and boundary element method is proposed. Issues related to the implementation, the accuracy and the generality of the method are discussed. The efficiency of the approach is illustrated by numerical results.

Key words. Bernoulli, free boundary, level-set, boundary elements

1. Introduction. Bernoulli free boundary problems find their origin in the description of free surfaces for ideal fluids [9]. There are, however, numerous other applications leading to similar formulations, see for instance [8]. For concreteness, we focus on the exterior Bernoulli problem. Let Ω be a bounded domain in $\mathbb{R}^2$. The exterior Bernoulli problem consists in seeking a bounded domain $A \supset \Omega$ and a function u defined on $\bar{A} \setminus \Omega$ such that :

$$\Delta u = 0 \qquad \text{in } A \setminus \overline{\Omega}, \tag{1.1}$$

$$u = 1 \qquad \text{on } \partial\Omega \tag{1.2}$$

$$u = 0 \qquad \text{on } \partial A, \tag{1.3}$$

$$\frac{\partial u}{\partial n} = \mu \qquad \text{on } \partial A, \tag{1.4}$$

where μ is given. In the previous example, one can think of u as a streamfunction and of Ω as an obstacle. Taking into account (1.3), condition (1.4) can be written as $|\nabla u| = |\mu|$ and corresponds, for fluid applications, to Bernoulli's principle, see for instance [6].

The above problem has been extensively studied, see [8] for general remarks. For a convex simply connected bounded domain Ω, it is known that for any negative constant $\mu < 0$, the above problem admits a unique classical solution. Further, the free boundary ∂A has regularity $\mathcal{C}^{2,\alpha}$, see [18], Theorem 1.1[1]. The convexity assumption is necessary for uniqueness as counterexamples show (see [8], Example 13). The study of the interior Bernoulli problem is more delicate and not even convexity can ensure uniqueness.

There are roughly two ways of tackling such problems numerically. First, a variational formulation may be considered and the corresponding cost function minimized [16, 20, 23]; this requires the calculations of shape gradients. Second, a fixed point type approach can be set up where a sequence of elliptic problems are solved in a sequence of converging domains, those domains being obtained through some updating

*Department of Mathematics and Center for Research in Scientific Computation, North Carolina State University, Raleigh, NC 27695-8205, USA (cmkuster@ncsu.edu). Partially supported by the National Science Foundation (NSF) through grant DMS-0244488.

†Department of Mathematics and Center for Research in Scientific Computation, North Carolina State University, Raleigh, NC 27695-8205, USA (gremaud@ncsu.edu). Partially supported by the National Science Foundation (NSF) through grants DMS-0204578, DMS-0244488 and DMS-0410561.

‡Laboratoire de Mathématiques, CNRS UMR 6620, Université Blaise Pascal (Clermont-Ferrand) 63177 Aubière cedex, France (Rachid.Touzani@math.univ-bpclermont.fr).

[1]The result is in fact established in the more general case of the p-Laplacian in [18] and is generalized to non-constant μ in [19].

rule at each iteration [5, 8, 21]. The method studied in this paper falls in the latter category.

More specifically, the strategy consists in solving the potential problem with one of the conditions on the free boundary omitted and then using the omitted condition to update the location of the free boundary. Given an initial domain $A_0 \supset \bar{\Omega}$, the simplest variant of this type consists in solving the sequence of problems

$$\Delta u_k = 0 \qquad \text{in } A_k \setminus \overline{\Omega}, k = 0, 1, 2, \ldots \tag{1.5}$$

$$u_k = 1 \qquad \text{on } \partial\Omega, \tag{1.6}$$

$$\partial_n u_k = \mu \qquad \text{on } \partial A_k. \tag{1.7}$$

For a given domain A_k, it is well known that problem (1.5–1.7) admits a unique solution, see for instance [12], Theorem 5.1. The new domain A_{k+1} is found by moving ∂A_k in its normal direction so that u_k vanishes there. Let $P_k \in \partial A_k$; to first order, we have

$$u_k(P_{k+1}) \approx u_k(P_k) + \mu\, d_k,$$

where $P_{k+1} = P_k + n_k d_k$, n_k being the outer unit normal to ∂A_k at P_k. The new point P_{k+1}, or similarly the distance d_k, is determined by the requirement $u_k(P_{k+1}) = 0$, i.e., $d_k = \frac{-u_k(P_k)}{\mu}$. The free boundary is thus updated according to[2]

$$\partial A_{k+1} = \partial A_k - \frac{u_k}{\mu}\, n_k. \tag{1.8}$$

The implementation of the above algorithm relies on two important numerical tools: first, the interface is represented through a level-set formulation, second, the elliptic problem (1.5–1.7) is solved through a boundary element method. Therefore the method requires, in principle, only the calculations of quantities being defined on the interface or close to it. Both level-set and boundary element methods are introduced and discussed in the present context in Section 2 and 3, respectively. The feasibility of the method is then tested in Section 4. Brief conclusions are offered in Section 5.

2. Level-set representation of the interface. In the above iterative process, the interface ∂A_k has to be updated as long as the residual $u_k|_{\partial A_k}$ is not (numerically) zero. Relation (1.8) indicates that the boundary should be moved in the normal direction by an amount proportional to the residual.

If the residual is considered as a normal speed, a time dependent problem can be set up for the "evolution" (or correction) of the domain A_k. We denote the domain so generated $A_k(t)$ with $A_k(0) = A_k$. Let $F : \mathbb{R}^2 \to \mathbb{R}$ be an extension of the residual away from A_k (see subsection 2.3 below). We want the interface $\Gamma(t) = \partial A_k(t)$ to be characterized by

$$\Gamma(0) = \partial A_k, \qquad \Gamma(t) = \{(x(t), t); x(0) = x_0, x_0 \in \partial A_k\} \quad \text{for } t > 0,$$

where

$$\frac{dx}{dt} = Fn, \qquad x(0) = x_0,$$

[2]The present iterative process can be modified by imposing the Neumann condition (1.7) on the "next" boundary ∂A_{k+1} instead of ∂A_k [8, 10]. However, in the formulation adopted here, those modifications are of little use as they require information such as curvature which complicates the calculation of the "evolution" of the interface.

n being the unit outer normal to $\partial A_k(t)$. A level-set approach, as pioneered in [25] (see also [24, 28]), consists in representing the interface $\{\Gamma(t)\}_{t\geq 0}$ as the zero level-set of family of level set functions $\{\phi(\cdot,t)\}_{t\geq 0}$ with the property

$$A_k(t) = \{x \in \mathbb{R}^2; \phi(x,t) < 0\}, \qquad \Gamma(t) = \partial A_k(t) = \{x \in \mathbb{R}^2; \phi(x,t) = 0\},$$

where $A_k(t)$ denotes the domain stemming from the evolution of A_k through the above process. By taking the time derivative of the relation $\phi(x(t),t) = 0$, the level-set equation is obtained

$$\partial_t \phi + F|\nabla \phi| = 0, \tag{2.1}$$

$$\phi(\cdot,0) = \phi_0, \tag{2.2}$$

where ϕ_0 is a level-set function corresponding to ∂A_k.

Several points related to the implementation of the above method have now to be considered. The fixed boundary $\partial\Omega$ is approximated by a piecewise linear curve $\partial\Omega_h$ with N elements. The size of the smallest element of $\partial\Omega_h$ is denoted Δx. Let $B \subset \mathbb{R}^2$ be a square domain of size $M\Delta x \times M\Delta x$ where M is chosen large enough so that B contains A. We associate to B, in a natural way, a uniform Cartesian mesh B_h of size Δx. In the following, the level-set functions are characterized by their nodal values on the mesh B_h. Two kinds of interpolation operators are considered on B_h. In the Contouring step (subsection 2.1), a classical $\mathcal{P}_1$ interpolation is used: each square cell is divided into two triangular elements[3] and on each of those triangles, the unique polynomial of degree 1 agreeing with the values of the level-set function at the vertices is constructed. For a given level-set function ϕ on B_h, the $\mathcal{P}_1$ interpolant of ϕ is denoted $I_{\mathcal{P}}\phi$. In the Projection step (subsection 2.2), a classical local $\mathcal{Q}_2$ interpolation is considered: to each node x_i, we associate its eight closest neighbors and construct on this set of nine nodes the unique polynomial of degree 2 in each variable agreeing with the values of ϕ there. This local $\mathcal{Q}_2$ interpolant is denoted $I_{\mathcal{Q},x_i}\phi$.

2.1. Contouring. In many applications of the level-set method, the actual reconstruction of the interfaces is not needed. This is not the case here as the elliptic problem (1.5–1.7) has to be solved in a family of successive domains defined by those interfaces. Let ϕ be a given level-set function and $I_{\mathcal{P}}\phi$ its $\mathcal{P}_1$-interpolant on B_h. By contouring, we mean the operation that associates to the nodal values of ϕ the zero level-set of $I_{\mathcal{P}}\phi$. This construction yields an outward normal to $A_k(t)$ since the gradient of $I_{\mathcal{P}}\phi$ is piecewise constant.

If $I_{\mathcal{P}}\phi$ is uniformly equal to zero on a given triangle, then the problem is under-resolved (Δx is too large) and the algorithm fails.

2.2. Projection. As mentioned above, some quantities such as the normal speed need to be extended away from the interface. The first step in this process consists again in an accurate reconstruction of the boundary. A loop through the mesh is done to determine which nodes are "close to" the interface. Here, a node is close to the interface if it has a primary neighbor where ϕ has opposite sign.

For each node x_i close to the interface, the closest point $x^\star$ to x_i on the zero level-set of $I_{\mathcal{Q},x_i}\phi$ is computed. The square of the Cartesian distance, i.e., $|x - x_i|^2$ is minimized subject to the constraint $I_{\mathcal{Q},x_i}\phi = 0$. The Lagrangian for this problem is

$$\mathcal{L}(x,\lambda) = |x - x_i|^2 + \lambda\, I_{\mathcal{Q},x_i}\phi(x). \tag{2.3}$$

[3] Any such decomposition is acceptable.

To find the point $x^\star$, the system

$$\nabla_{x,\lambda}\mathcal{L}(x,\lambda) = 0, \tag{2.4}$$

is solved by Newton's method with Armijo line search [22]. This projection method was introduced and discussed in [14]. This projection step could potentially be used to reconstruct the interface, i.e., as another Contouring step. However, while it is locally more accurate than the above contouring algorithm (and is thus ideal when used in conjunction with the Extension step described below), it has several disadvantages as a contouring tool (local character, non constant element-wise gradient).

2.3. Extension. This step extends the speed F away from the interface, so that (2.1, 2.2) can be solved. Let Γ be the interface obtained from the Contouring step. By construction, Γ is a closed (for the problems considered here), possibly multi-connected, piecewise linear (on the triangular mesh derived from B_h) curve in B. The boundary nodes corresponding to Γ are denoted $\{\xi_j\}$, i.e., the ξ_j's are the end points of the line segments that form Γ. Further, through the elliptic step (see Section 3), the value of F at the center point of each linear segment of Γ is known; the center point nodes are denoted $\{\bar{\xi}_j\}$.

First, the values of F at the midpoint boundary nodes $\{\bar{\xi}_j\}$ are extended to the set of Cartesian nodes $\{x_i\}$ that form the vertices of the triangles containing the nodes $\{\bar{\xi}_j\}$.

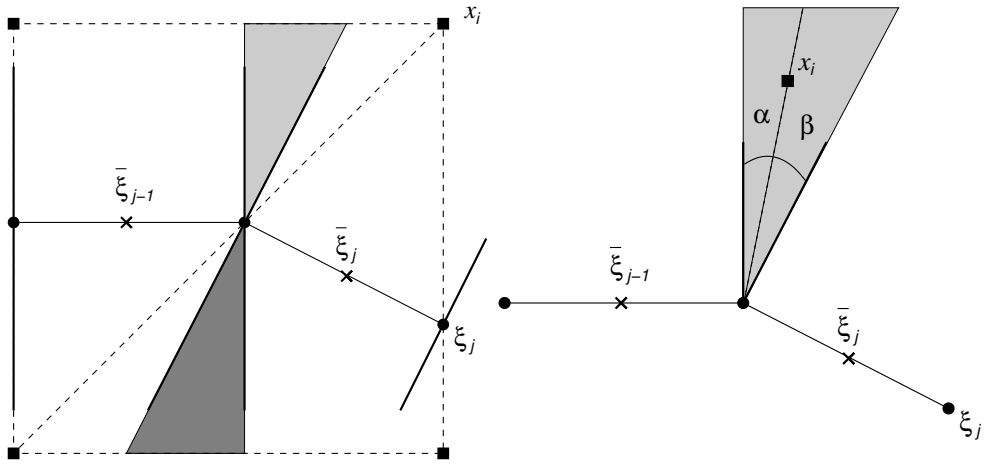

FIG. 2.1. Left: general view of the geometry involved in the local Extension step and corresponding "domains of influence", dark grey area: shock-like domain, light grey area: rarefaction-like domain; right: definition of the local angles α and β.

More precisely, consider the node $\bar{\xi}_j$ in Figure 2.1, left. This local extension step is based on the analysis of the "domain of influence" of the nodes $\{\bar{\xi}_j\}$. This domain of influence is taken here as the set of all the points in B whose orthogonal projection on the line containing the segment through $\bar{\xi}_j$ belongs to that segment. If a node x_i belongs to the domain of influence of $\bar{\xi}_j$, then F is extended at x_i by the value $F(\bar{\xi}_j)$ as in Figure 2.1, left. If instead the node x_i is in the domain of influence of more than one midpoint boundary node, as would be the case in the dark grey area in Figure 2.1, left, then the value at the midpoint of the closest segment to x_i is retained. Finally, if x_i does not belong to the domain of influence of any midpoint boundary node, as

in the light grey area in Figure 2.1, right, then the value of F at x_i is taken as

$$F(x_j) = \frac{\beta}{\alpha+\beta} F_{j-1} + \frac{\alpha}{\alpha+\beta} F_j,$$

where F_{j-1} and F_j are the values of F at the midpoint boundary nodes $\bar{\xi}_{j-1}$ and $\bar{\xi}_j$ respectively and where the angles α and β are defined as in Figure 2.1, right. This way of defining the local extension of F is compatible with the global extension (2.7).

Second, a renormalized function $\tilde{\phi}$ is initialized as a signed distance function at the same Cartesian nodes at which F has just been extended. The Projection step is used to do this.

We emphasize that both of those local extension steps for F and $\tilde{\phi}$ only take place on the nodes adjacent to the interface; the corresponding values are then used as starting points for the extension to the rest of the Cartesian nodes. This is accomplished using the Fast Marching method [2] to solve

$$|\nabla \tilde{\phi}| = 1 \qquad \text{in } B, \tag{2.5}$$
$$\tilde{\phi} = 0 \qquad \text{on } \Gamma, \tag{2.6}$$
$$\nabla \tilde{F} \cdot \nabla \tilde{\phi} = 0 \qquad \text{in } B, \tag{2.7}$$
$$\tilde{F} = F \qquad \text{on } \Gamma. \tag{2.8}$$

A fully upwind mixed first/second order discretization of the above equations is applied on the mesh B_h, see [15, 29, 30] for more details.

2.4. Updating the interface. The interface is moved by updating the corresponding level-set function through (2.1, 2.2). More precisely, after the Extension step, the level-set function $\tilde{\phi}$ corresponding to the current interface Γ is a signed distance function and in particular, $|\nabla\tilde{\phi}| = 1$. Therefore, (2.1, 2.2) reads here

$$\partial_t \phi + \tilde{F} = 0,$$
$$\phi(\cdot, 0) = \tilde{\phi}.$$

The update is then trivially computed by taking one forward Euler step

$$\phi_{new} = \tilde{\phi} - \Delta t\, \tilde{F}, \tag{2.9}$$

where $\Delta t = -\frac{1}{2\mu}$. Note that this corresponds to half the optimal time step given by (1.8); taking the "optimal" value from (1.8) may lead to overshoots in the position of the interface and may in fact result in slowing down the convergence of the global iterative process.

2.5. Initial interface. An initial guess of the interface's position, ∂A_0, needs to be provided. This can be done in an ad hoc way. In Section 4, ∂A_0 is taken as a curve of constant distance to Ω.

3. Boundary element method. Consider again the problem (1.5-1.7). For the sake of simplicity, the subscript k is dropped in this section. We assume both ∂A and $\partial\Omega$ to be simple closed curves and let $\Gamma = \partial A \cup \partial\Omega$. The region of interest $A \setminus \bar{\Omega}$ being interior to ∂A and exterior to $\partial\Omega$, ∂A is oriented counterclockwise while $\partial\Omega$ is clockwise.

Multiplying (1.5) by the fundamental solution $G(x,y) := -\frac{1}{2\pi}\log|x-y|$ and integrating twice by parts leads to

$$u(x) = \int_\Gamma G(x,y)\, \frac{\partial u}{\partial n_y}(y)\, ds(y) - \int_\Gamma \frac{\partial G}{\partial n_y}(x,y)\, u(y)\, ds(y), \tag{3.1}$$

where n is the unit outer normal to $A \setminus \bar{\Omega}$ at y. The above integral representation is valid for $x \in A \setminus \bar{\Omega}$. To treat the case $x \in \Gamma$, we define the linear operator $\mathcal{L} : L^2(\Gamma) \to L^2(\Gamma)$ by

$$\mathcal{L}v(x) = \begin{cases} \frac{v(x)}{2} + \int_{\partial A} \frac{\partial G}{\partial n_y}(x,y)\, v(y)\, ds(y) - \int_{\partial\Omega} G(x,y)\, v(y)\, ds(y) & \text{for } x \in \partial A, \\ \int_{\partial A} \frac{\partial G}{\partial n_y}(x,y)\, v(y)\, ds(y) - \int_{\partial\Omega} G(x,y)\, v(y)\, ds(y) & \text{for } x \in \partial\Omega, \end{cases}$$

and the function $\mathcal{F} \in L^2(\Gamma)$

$$\mathcal{F}(x) = \begin{cases} \mu \int_{\partial A} G(x,y)\, ds(y) - \int_{\partial\Omega} \frac{\partial G}{\partial n_y}(x,y)\, ds(y) & \text{for } x \in \partial A, \\ \mu \int_{\partial A} G(x,y)\, ds(y) - \int_{\partial\Omega} \frac{\partial G}{\partial n_y}(x,y)\, ds(y) - \frac{1}{2} & \text{for } x \in \partial\Omega. \end{cases}$$

Taking into account the boundary conditions (1.6) and (1.7), it is then standard to check that if

$$w(x) = \begin{cases} u(x) & \text{for } x \in \partial A, \\ \frac{\partial u}{\partial n}(x) & \text{for } x \in \partial\Omega, \end{cases}$$

where u is the solution to (1.5-1.7) then

$$\mathcal{L}w(x) = \mathcal{F}(x), \qquad \forall x \in \Gamma. \tag{3.2}$$

Problem (3.2) is discretized as follows. The interface $\partial A = \partial A_h$ is obtained through contouring of a given level-set function, see Section 2 and piecewise constant elements are considered. The function w solution to (3.2) is approximated by w_h such that

$$w_h(x) = w_e \qquad \forall x \in e,$$

where e is an edge of either ∂A_h or $\partial\Omega_h$. Equation (3.2) is then collocated at the midpoints of the edges. In other words, for a generic piecewise constant function v_h, we define

$$\mathcal{L}_h v_h(\bar{\xi}_e) = \begin{cases} \frac{v_e}{2} + \int_{\partial A_h} \frac{\partial G}{\partial n_y}(\bar{\xi}_e, y)\, v_h(y)\, ds(y) - \int_{\partial\Omega_h} G(\bar{\xi}_e, y)\, v_h(y)\, ds(y) & \text{for } \bar{\xi}_e \in \partial A_h, \\ \int_{\partial A_h} \frac{\partial G}{\partial n_y}(\bar{\xi}_e, y)\, v_h(y)\, ds(y) - \int_{\partial\Omega_h} G(\bar{\xi}_e, y)\, v_h(y)\, ds(y) & \text{for } \bar{\xi}_e \in \partial\Omega_h, \end{cases}$$

where $\bar{\xi}_e$ is the midpoint of the edge e. Similarly, we also have

$$\mathcal{F}_h(\bar{\xi}_e) = \begin{cases} \mu \int_{\partial A_h} G(\bar{\xi}_e, y)\, ds(y) - \int_{\partial\Omega_h} \frac{\partial G}{\partial n_y}(\bar{\xi}_e, y)\, ds(y) & \text{for } \bar{\xi}_e \in \partial A_h, \\ \mu \int_{\partial A_h} G(\bar{\xi}_e, y)\, ds(y) - \int_{\partial\Omega_h} \frac{\partial G}{\partial n_y}(\bar{\xi}_e, y)\, ds(y) - \frac{1}{2} & \text{for } \bar{\xi}_e \in \partial\Omega_h. \end{cases}$$

The approximate solution w_h is the solution to

$$\mathcal{L}_h w_h(\bar{\xi}_e) = \mathcal{F}_h(\bar{\xi}_e), \qquad \forall \bar{\xi}_e \in \partial A_h \cup \partial\Omega_h. \tag{3.3}$$

The above integrals are computed exactly in the present implementation.

Both the integral equation (3.2) and the linear problem (3.3) are well conditioned. It can be verified that $\mathcal{L}_h$ admits eigenvalues and singular values that are bounded independent of the mesh, see e.g. [17] or [26] for explicit expressions of the eigenvalues in some specific cases. Further, in spite of the fact that the elements of ∂A_h are allowed

to be arbitrarily small, the condition number of $\mathcal{L}_h$ has been numerically verified to be of order N which would correspond to the uniform mesh case [3]. The condition number of the matrices corresponding to the numerical tests of Section 4 are on the order of 100. The resulting linear system is solved by GMRES [22, 27] which is consequently expected to perform well here even without preconditioning. GMRES is restarted after 20 steps (i.e., the solver is GMRES(20)) and is stopped on small relative residuals, more precisely the stopping criterion is

$$\|\mathcal{F}_h - \mathcal{L}_h w\|_2 \leq 10^{-10} \|\mathcal{F}_h\|_2,$$

where w denotes the current iterate. With the above parameters, GMRES has been observed to perform slightly better than other CG-like methods such as QMR and Bi-CGSTAB [22] on the test problems of Section 4.

4. Numerical results.

4.1. Algorithm. To solve the External Bernoulli problem, we use the following algorithm:

Input: a discretization of the boundary $\partial\Omega$, $\partial\Omega_h$, μ
Create the underlying Cartesian grid B_h
Create a level set function ϕ on B_h corresponding to ∂A_0
$k = 0$
$R_{-1} = 10^{10}$ (initial residual)
loop
  Contour ϕ (subsection 2.1) to find ∂A_k
  Solve (3.3) to get u
  $R_k = \max$ of $|u|$ on ∂A_k
  if $(R_{k-1} - R_k)/R_k < 10^{-3}$ (small residual decrease) **then**
    STOP
  end if
  Set $F = u$ on ∂A_k
  Extend ϕ and F (subsections 2.2 and 2.3)
  Move boundary (subsection 2.4)
  $k = k + 1$
end loop

Several remarks are in order.

- Progressive mesh refinement can be considered, i.e., a coarse mesh solution can be used as starting point. A strategy of this type is for instance used in [20] for a similar type of problems but for a different numerical approach.
- In solving (3.3), the "missing" condition (1.3) can be used when choosing the initial iterate for GMRES. This results in faster convergence (less GMRES iterates) as the algorithm progresses.
- The extension step through Fast Marching (Subsection 2.3) is done in the whole computational domain B. The corresponding complexity is $\mathcal{O}(M^2 \log M)$ where M^2 is the total number of nodes in the Cartesian grid B_h. A narrow band implementation [2, 31] could be considered to speed up the algorithm. However, the global complexity of the problem would not change since, if the width of the band is a constant multiple of Δx, say $n\Delta x$, then by (2.9), Δt should be reduced from $\frac{-1}{2\mu}$ to a value less than $\frac{n\Delta x}{|\vec{F}|}$ since the band has to con-

tain the boundary. Fast summation techniques can also be implemented [13] to bring down the cost of solving the linear system (3.3), which accounts for most of the computational cost, from $\mathcal{O}(N^2)$ with the present implementation to $\mathcal{O}(N)$, where N is the number of elements of $\partial\Omega_h$. A quasi-optimal global complexity of $\mathcal{O}(M^2) = \mathcal{O}(N^2)$ is computationally observed in Section 4.

- Higher order boundary element methods can be used [4]. Second order convergence is observed in Section 4 (partially as a result of the solution being constant on the outer free boundary). To the authors' knowledge, the present work is one of very few published results regarding the accuracy of a combined level-set boundary element method, see for instance [11].

4.2. Example 1. Following [8], a quick look at the radial case is instructive. Let Ω be the unit ball. We consider the problem (1.1-1.4) with Ω as above and $\mu = -2$. The solution to (1.1-1.3) with A being the ball of radius R centered at the origin is

$$u(r) = -2\,R\log r + 1,$$

expressed in polar coordinates. An iterative process similar to the one above can then be considered. Taking (1.8) into account, the k-th step of the algorithm reads

$$R_{k+1} = R_k - R_k\,\log R_k + \frac{1}{2}, \qquad k = 1, 2, \ldots$$

Therefore in the fully radial case, the problem amounts to finding a fixed point to the function $f(R) = R - R\log R + \frac{1}{2}$. The function f has a unique fixed point $\bar{R}$ where

$$\bar{R} = \frac{1}{2\,W(\frac{1}{2})},$$

the function W being the Lambert W function[4] [7].

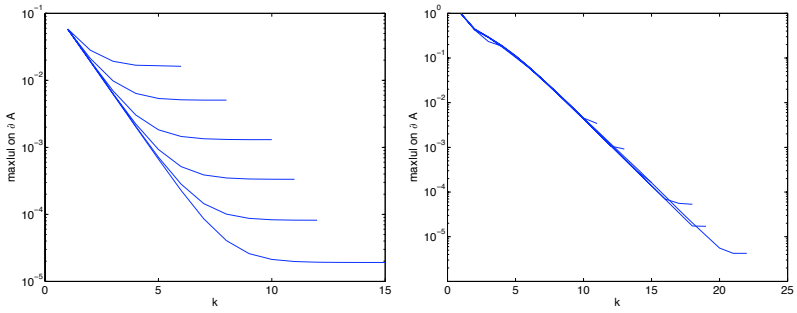

FIG. 4.1. Convergence history of the iterations: maximum value of $|u|$ on the current interface as a function of the iteration number; left: example 1, right: example 2. The convergence curves are in the obvious order: from top to bottom, $N = 25, 50, 100, 200, 400, 800$.

Table 4.1 shows second order convergence in the L^∞ norm. Full second order convergence is obtained even though the elliptic solver is based on a piecewise constant discretization. This is due to the fact that the exact solution is constant on both the inner and outer boundaries. Further, when measured with respect to runtime, the complexity is also second order.

[4]The Lambert W function is the inverse of $W \mapsto z = W\,e^W$.

N	$\max \lvert u \rvert$ on Γ	Rate	L^∞ Error	Rate	k	Time	Rate
25	1.64(-2)	–	1.53(-2)	–	6	0.71	–
50	5.07(-3)	1.7	4.40(-3)	1.8	8	4.1	2.5
100	1.30(-3)	2.0	1.12(-3)	2.0	10	19	2.2
200	3.34(-4)	2.0	2.85(-4)	2.0	11	76	2.0
400	8.18(-5)	2.0	7.10(-5)	2.0	12	372	2.1
800	1.92(-5)	2.1	1.74(-5)	2.0	14	1465	2.0

TABLE 4.1
Convergence and complexity rates for Example 1 (radial case); N: number of elements on $\partial\Omega_h$, L^∞ Error refers to the Hausdorff distance between exact and computed boundaries, k is the number of nonlinear iterations (see Section 4.1), Time is the runtime in seconds.

The convergence history is instructive. Figure 4.1, left, displays the error (maximum of $|u|$ on the free boundary) through the iterations. The behavior of the first iterates is governed by the geometry, see (2.9), and is only weakly dependent on the mesh size Δx. The later iterations during which the fine structure of the boundary is determined do depend on Δx. This explains the mesh dependency of the number of iterations observed in Table 4.1.

4.3. Example 2. We consider here the problem (1.1-1.4) with Ω consisting of two disks of radius 1, one centered at (-2,2), the one at (2,-2); further, $\mu = -1/4$. The initial boundary is taken as two circles of radius 1.1, one around each of the inner disks. Note that for this choice of μ, the exact boundary is simply connected. A couple of iterates are displayed in Figure 4.2. One can note that after the first step already the correct topology of the interface has been achieved.

No exact solution is available for the present example. In Table 4.2, the maximum of u on the boundary is reported. By construction, this maximum should vanish for the converged solution. The complexity, as measured from the runtimes, is also reported. In both cases, the rates are about two.

N	$\max \lvert u \rvert$ on Γ	Rate	k	Time	Rate
26	3.43(-3)	–	11	4.4	–
50	9.16(-4)	2.0	13	19	2.2
100	1.64(-4)	2.5	15	85	2.2
200	5.31(-5)	1.6	18	573	2.8
400	1.71(-5)	1.6	19	1678	1.6
800	4.25(-6)	2.0	21	7613	2.2

TABLE 4.2
Convergence and complexity rates for Example 2; N: number of elements on $\partial\Omega_h$, k is the number of nonlinear iterations (see Section 4.1), Time is the runtime in seconds.

Convergence history is displayed in Figure 4.1, right; a behavior similar as that of Example 1 is observed.

5. Conclusion. Solutions of the Bernoulli free boundary problem can be efficiently computed by the method presented here. Providing a Green's function is available, the method can be used to solve other free boundary problems. For instance, it can be applied with only minor modifications to the Prandtl-Batchelor problem (see

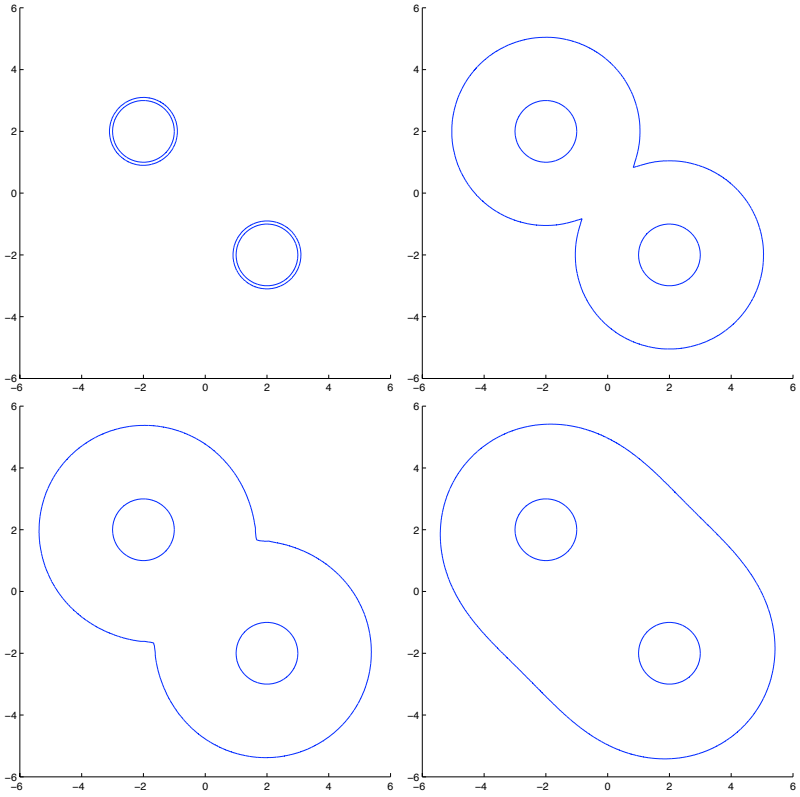

FIG. 4.2. Evolution of the interface for Example 2 at the initial step and after steps 1, 2 and 15 ($N = 100$).

[1] and the references therein) which consists in looking for a domain A which is now interior to the fixed domain Ω such that for a given function σ

$$|\nabla u_A|^2 - |\nabla u_\Omega|^2 = \sigma, \qquad \text{on } \partial A,$$

where u_A and u_Ω solve

$$\begin{cases} \Delta u_A = -\omega & \text{in } A, \\ u_A = 0 & \text{on } \partial A, \end{cases} \qquad \begin{cases} \Delta u_\Omega = 0 & \text{in } \Omega, \\ u_\Omega = 0 & \text{on } \partial A, \\ u_\Omega = \mu & \text{on } \partial\Omega, \end{cases}$$

μ and ω being positive constants.

REFERENCES

[1] A. ACKER, *On the existence of convex solutions to a generalized Prandtl-Batchelor free boundary problem-II*, ZAMP, 53 (2002), pp. 438–485.

[2] D. ADALSTEINSSON AND J.A. SETHIAN, *The fast construction of extension velocities in level set methods*, J. Comput. Phys., 148 (1999), pp. 2–22.

[3] M. AINSWORTH, W. MCLEAN AND T. TRAN, *The conditioning of boundary element equations on locally refined meshes and preconditioning by diagonal scaling*, SIAM J. Numer. Anal., 36 (1999), pp. 1901–1932.

[4] K.E. ATKINSON, *The numerical solutions of integral equations of the second kind*, Cambridge Monograph in Applied Mathematics, #4, 1997.

[5] F. Bouchon, S. Clain and R. Touzani, *Numerical solution of the free boundary Bernoulli problem using a level set formulation*, Comp. Meth. Appl. Mech. and Eng., 194 (2005), pp. 3934-3948.
[6] A.J. Chorin and J.E. Marsden, *A mathematical introduction to fluid mechanics*, Springer, 1992.
[7] R.M. Corless, G.H. Gonnet, D.E.G. Hare, D.J. Jeffrey and D.E. Knuth, *On the Lambert W function*, Adv. Comput. Math., 5 (1996), pp. 329–359.
[8] M. Flucher and M. Rumpf, *Bernoulli's free-boundary problem, Qualitative theory and numerical approximation*, J. Reine Angew. Math., 486 (1997), pp. 165–204.
[9] K.O. Friedrichs, *Über ein Minimumproblem für Potentialströmungen mit freiem Rand*, Math. Ann., 109 (1934), pp. 60–82.
[10] P.R. Garabedian, *Calculation of axially symmetric cavities and jets*, Pacific J. Math., 6 (1956), pp 611-684.
[11] M. Garzon, D. Adalsteinsson, L. Gray and J.A. Sethian, *A coupled level set-boundary integral method for moving boundary simulations*, Interfaces Free Bound., 7 (2005), pp. 277–302.
[12] R. González and R. Kress, *On the treatment of a Dirichlet-Neumann mixed boundary value problem for harmonic functions by an integral equation method*, SIAM J. Math. Anal., 8 (1977), pp 504–517.
[13] A. Greenbaum, L. Greengard and G.B. McFadden, *Laplace's equation and the Dirichlet-Neumann map in multiply connected domains*, J. Comput. Phys, 105 (1993), pp. 267–278.
[14] P.A. Gremaud, C.M. Kuster and Zhilin Li, *A Study of Numerical Methods for the Level Set Approach*, NCSU-CRSC Tech Report CRSC-TR05-39, submitted to Appl. Num. Math.
[15] P.A. Gremaud and C.M. Kuster, *Computational study of fast methods for the Eikonal equation*, SIAM J. Sci. Comput, 27 (2006), pp. 1803-1816.
[16] J. Haslinger, T. Kozubek, K. Kunisch and G. Peichl, *Shape optimization and ficticious domain approach for solving free-boundary problems of Bernoulli type*, Comput. Optim. Appl., 26 (2003), pp. 231–251.
[17] J. Hayes and R. Kellner, *The eigenvalue problem for a pair of coupled integral equations arising in the numerical solution of Laplace's equation*, SIAM J. Appl. Math., 22 (1972), pp. 503–513.
[18] A. Henrot and H. Shahgholian, *Existence of classical solutions to a free boundary problem for the p-Laplace operator: (I) the exterior convex case*, J. reine angew. Math., 521 (2000), pp. 85–97.
[19] A. Henrot and H. Shahgholian, *The one phase free boundary problem for the p-laplacian with non-constant Bernoulli boundary condition*, Trans. Amer. Math. Soc., 354 (2002), pp. 2399–2416.
[20] K. Ito, K. Kunisch and G.H. Peichl, *Variational approach to shape derivatives for a class of Bernoulli problems*, J. Math. Anal. Appl. 314 (2006), pp. 126–149.
[21] K.T. Kärkkäinen and T. Tiihonen, *Free surfaces: shape sensitivity analysis and numerical methods*, Int. J. Numer. Engng., 44 (1999), pp. 1079–1098.
[22] C.T. Kelley, *Iterative methods for linear and nonlinear problems*, Frontiers in Applied Mathematics #16, SIAM, 1995.
[23] G. Mejak, *Numerical solution of Bernoulli-type free boundary value problems by variable domain method*, Int. J. Numer. Meth. Engng., 37 (1994), pp. 4219–4245.
[24] S.J. Osher and R.P. Fedkiw, *Level set methods and dynamic implicit surfaces*, Applied Mathematical Sciences #153, Springer, 2002.
[25] S. Osher and J. Sethian, *Fronts propagating with curvature dependent speed: algorithms based on Hamilton-Jacobi formulations*, J. Comput. Phys., 56 (1988), pp. 12–49.
[26] G.J. Rodin and O. Steinbach, *Boundary element preconditoners for problems defined in slender domains*, SIAM J. Sci. Comput., 24 (2003), pp. 1450–1464.
[27] Y. Saad and M. Schultz, *GMRES a generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 856–869.
[28] J.A. Sethian, *Level Set Methods and Fast Marching Methods*, Cambridge University Press (1999).
[29] J.A. Sethian and A. Vladimirsky, *Fast methods for the Eikonal and related Hamilton-Jacobi equations on unstructured meshes*, Proc. Natl. Acad. Sci. USA, 97 (2000), pp. 5699–5703.
[30] J.A. Sethian and A. Vladimirsky, *Ordered upwind methods for static Hamilton-Jacobi equations: theory and algorithms*, SIAM J. Numer. Anal., 41 (2003), pp. 325–363.
[31] L. Yatziv, A. Bartesaghi and G. Sapiro, *O(N) implementation of the fast marching algorithm*, J. Comput. Phys., 212 (2006), pp. 393-399.